SOFTWARE FOR AUTOMATIC MODEL SIMULATION IMPLEMENTATION

Ondřej Bartík

Doctoral Degree Programme (1), FEEC BUT

E-mail: xbarti07@stud.feec.vutbr.cz

Supervised by: Petr Blaha

E-mail: blahap@feec.vutbr.cz

Abstract: In the industry, it is always a problem to run an algorithm on the real device because, of the possible unpredictable behavior. This behavior could lead to the temporary malfunction of the device or, in the worst scenario, to the permanent damage of the device. Because of this, it is necessary to perform a simulation with suitable model of the device, which reflects the reality as much as possible. The way in which the simulation is done needs to be considered as well. Model In the Loop (MIL) simulation approach and Hardware In the Loop (HIL) are suitable to test the whole system before it is commissioned. The real time operateability of the models is necessary to run these types of simulations. A suitable platform and form of the models need to be chosen for these purposes. OpenModelica provides wide range of possibilities to create desired model and also provides the possibility to export model in Functional Mock Up (FMU) format. The NI Compact Rio platform provides many options with customization due to the FPGA and real-time Linux based operating system. As it is, NI CompactRIO is suitable platform for MIL/HIL model implementation.

Keywords: OpenModelica, FMU, NI CompactRIO, Autogenerate simulation

1 INTRODUCTION

The industry of these days aims to best financial efficiency, shortest time-to-market and the safety of all processes. Initiative industry 4.0 works with tools as digital twins and digital factory to suppress as many hazards as possible and to increase efficiency of all process in the industry. The MIL and HIL simulations have their place among those tools where HIL simulations become more preferred in these days [1], [2]. It is more reliable, cost and time effective then its virtual alternative Model In the Loop (MIL), which is the scenario that doesn't evaluate any data exchange based on the sensors and actuators, which are used in real device. MIL based simulation requires real time operation of the model and system and appropriate link connection between each block of simulation [3]. The HIL based simulation could use the same model as MIL based scenario, but in the case of the HIL, the model is supplemented by appropriate sensors and actuators to get it as close as possible to the comparison with the real system.

The correct choice of the model is always a tradeoff between complexity of the model and its accuracy. But the care of which platform to create a model and how it supposes to be migrated to the target platform should be taken into account as well. Today market is full of tools for modeling the dynamic systems. For instance: MATLAB/Simulink environment, OpenModelica, and Dymola. Last two mentioned are based on Modelica language, which is language to describe a mathematical and physical behavior of the dynamic systems. Modelica language allows create the acausal models. Where acausality in shell means thus signals can flow both direction from the input to the output and vice versa. With this functionality, it is possible to capture the behavior of the real electric machine, motor. Where electric motor can operate as energy converter in two operation modes. First is to transform the electrical energy to the mechanical energy (motor mode) and second is to transform the mechanical energy to the electrical energy (generator mode). Both of these operations

modes can be captured within a single simple model created with Modelica language. For purposes of this paper, OpenModelica Connection Editor environment [4] is chosen. Very important feature of OpenModelica Connection Editor is exporting the model in FMU format. More reading about FMU features, possibilities and structure is in the appropriate section *FMU Structure*.

National Instruments CompactRIO [5] is platform based on FPGA and processor based on ARMv7 architecture with real-time operating Linux based system (RTOS). The platform has wide connectivity option via NI plug-in cards and modules. The purpose of this platform is use the processor with RTOS to run model in real-time, which is the condition of MIL, respectively HIL simulations.

The main aim of this paper is to introduce the software based solution to autogenerate executable simulation with model instantiated on the NI CompactRIO platform. The model is in the FMU form exported from OpenModelica Connection Editor and the co-simulation mode of FMU is used to use the solver provided by OpenModelica environment.

The paper is organized as follows. Firstly, there is description of the FMU file in the section *FMU Structure*. Its structure, features are discussed in this section. Next, there is *Software for automatic FMU model implementation* section, which describes the issue of the model automatic implementation.

2 FMU STRUCTURE

FMU is open standard for model exchange among several custom simulation platforms [6]. The FMU file itself is an archive containing model description in *.xml file format, C sources files of the model, and compiled dynamic link libraries (*.dll, *.so). The model encapsulated in the FMU file is accessed by FMU interface (FMI) which is standardized. The interface is used for the simulation control (fetch inputs, do step, get outputs, etc.). It depends on how the FMU was exported from Modelica environment. In general, there are two approaches. First is FMU for model exchange only, that means thus only model was exported and if it contains any differential equations, or any other calculation (integration mainly), there need to be provided an external solver to handle this. Second approach is co-simulation. This means that model is encapsulated with the solver and FMI interface is mean only to control the simulation.

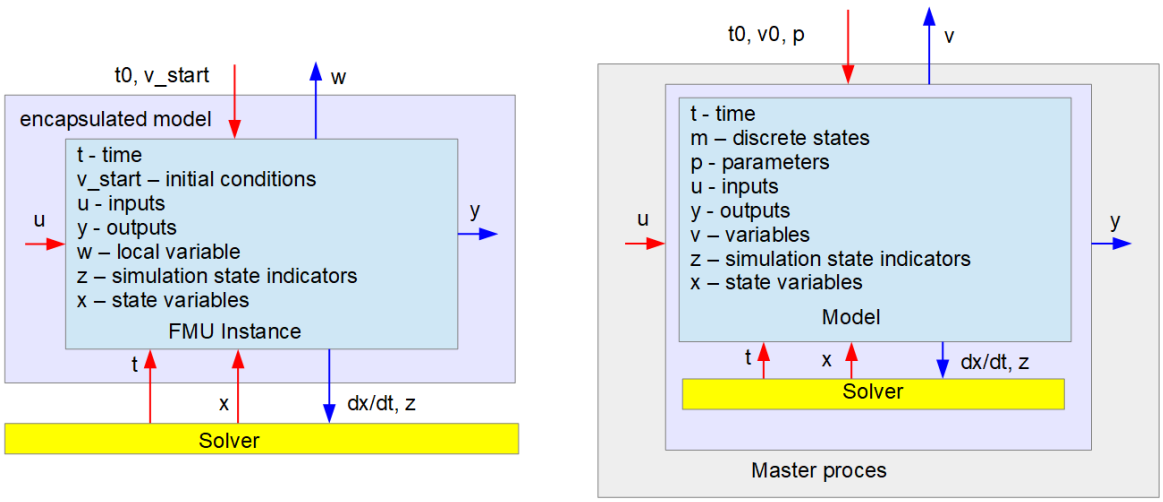

Figure 1: Model exchange model structure (left) and co-simulation model structure (right)

As mentioned above, FMU file is archive containing *.xml description file and source files. Description file contains GUID identification number for model handshake (instance creation), the name of the model, number of inputs and outputs of the model, number of internal variables and

states of the model. Another important information stored in the *.xml file is time horizon of the simulation and simulation step size (sampling period). All this information is defined during the model creation in the OpenModelica Connection Editor environment. Sources files could be split into two groups. The first one contains sources of the model itself. One of these is *<name>_FMU.c*, which creates an instance of the model. It also includes functions for parameters and variables handshake and ensures interconnection of the model source files with the FMI sources. As it was mentioned before, FMI ensures control of the model simulation, custom model parameter configuration, time horizon settings, and etc. The second group of source files are independent sources on the model and common for any model implemented, encapsulated in the FMU file. This group includes FMI sources, solver sources, additional functions for memory management and so on.

In the FMU archive, there is also a compiled model in dynamic link library form. That means, that there is *.dll or *.so file present, dependently on which platform (Windows or Linux) was the OpenModelica Connection Editor environment installed and FMU exported. Anyway, this library is compiled for x86/x64 platform and the desired platform is base on ARMv7 architecture, hence there is no importnat attention payed on this file within this paper.

It is important to mention, that the model, as it is exported in the FMU format, cannot run as a bare metal application on any platform. But the implementation expects some, kind of the operational system executing it.

Another important note, which must be kept on mind is, that OpenModelica environment has several restrictions of the model export. If the use of the Co-simulation is desired (which in this paper is), only FMU format version 2.0 could be used. Another restriction is thus OpenModelica Connection editor environment provides only Forward Euler solver for Co-simulation mode.

3 SOFTWARE FOR AUTOMATIC FMU MODEL IMPLEMENTATION

In order to get the model running on the NI CompactRIO platform, all the sources must be recompiled for the target ARMv7 architecture. There are two ways how to do it. The first one is to move all the sources from x86/x64 platform, where the model was created and FMU exported, onto the platform and run the compilation there. Then the second way is to use of the Cross-Compiler and stay on the x86/x64 platform, where the model was created and FMU file exported and run the compilation there. This platform was in the case of this paper a virtual PC with Ubuntu Linux distribution. To ease the recompilation as much as possible, there is an autogenerated makefile through the configuration script provided by OpenModelica Connection Editor environment. Unfortunately, the configure script and makefile are not ready for the Windows platform yet. This is the reason why the model creation and model recompilation must be done on the Linux platform. Another problem is caused by the National Instruments hardware driver support for Linux based systems. There is no driver software for Linux based systems. Due to these inconveniences, there must be switching between platforms (Linux/Windows). But the makefile support for Windwos from OpenModelica environment should be fixed in the future and hence, get rid of the necessity to recompile the model sources on the Linux platform. Once, when the sources are recompiled for ARMv7 architecture into the *.so file, there needs to a Master process be added to handle the simulation and the model. See Figure 1 for better view. The master process must get all necessary information from the *.xml file. The master process source is a template with the state machine inside. State machine is for the simulation handling. All information from the *.xml file is parsed into this template to create compilable source.

In order, to make these steps automatic, there are two scripts to handle this. The first one is aimed for the Linux platform and there for, it is a shell script *.sh. The purpose of the script is only to prepare makefile (define the compiler path, library extension, etc.), extract the source files from FMU archive, and run compilation. Then the compiled sources altogether with the *.xml file need to be moved onto the windows platform. The second script, extracts the information from the *.xml

file (name, number of the inputs and the outputs of the model, number of inner variables inside the model, etc.). Next the script modifies the master process template source and then, runs the compilation via the same Cross-Compiler, as the one used for the model recompilation on the Linux platform. Next operation done by script is opening the SSH communication channel and move the final application onto the NI CompactRIO and run it. Simulation results are stored on the CompactRIO and after finishing the simulation, it is moved through SSH to the *.log file, situated on the host PC. Figure 2 highlights the procedure.

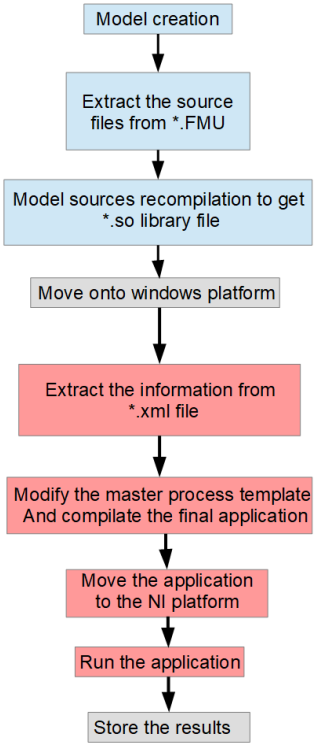

Figure 2: Automatic model implementation procedure

For better understanding, the blue blocks represent part which is done on the Linux platform and the red blocks represent part which is done on the Windows platform. Extra attention must be paid to step *Model creation*. It very depends on which Platform (Windows or Linux) is this step done. Because in dependency on the chosen platform, the appropriate makefile is enclosed to the FMU. On the other hand, once the model sources are recompiled into the dynamic link library form (*.so or *.dll), the final application (*Modify the master process template and compilate the final application* step) can be created

and compilated on both platforms (Windows, Linux). The compilation is platform independent process because of use the National Instruments Cross compilator. But for any operation with CompactRIO platform (SSH connection excluded) National Instruments drivers and hence, the Windows platform needs to be used.

4 CONCLUSION

This paper presented a way how to approach to the automatic model implementation using the model from OpenModelica Connection editor environment. Using the FMU standard and Co-simulation mode with solver included. Approach presented in this paper uses the NI CompactRIO as a target platform with ARMv7 architecture and real-time operation system on it. With the real-time functionality, the model implementation could be used as a basic element for the HIL simulation. The big advantage of this approach, thus it uses the OpenModelica environment, which is free and hence, no extra resources need to be spent on the modeling tools. Unfortunately, due the problem with the makefiles for custom recompilation of the models exported from OpenModelica environment, it is impossible to recompile the models on the Windows platform and Linux based system needs to be used. On the other hand, National Instruments does not officially provides any driver for its hardware for the Linux based platform. This platform inconsistency leads to the fact, that whole procedure cannot be done automatic and in certain point, the whole job must be moved to the different platform by human. This problem should be fixed by providing the finished makefiles for model sources recompilation by OpenModelica. For now, only elementary test tamplete is used. The template only runs the simulation as it is with no extra post parameter changing (change of time horizon for instance). The goal of the future work is create a modular-like template to give more options to run and control the simulation of the model.

ACKNOWLEDGEMENT

The completion of this paper was made possible by the grant No. FEKT-S-17-4234 - „Industry 4.0 in automation and cybernetics" financially supported by the Internal science fund of Brno University of Technology.

REFERENCES

[1] TAVANA, Nariman Roshandel and Venkata DINAVAHI. Real-Time Nonlinear Magnetic Equivalent Circuit Model of Induction Machine on FPGA for Hardware-in-the-Loop Simulation. *IEEE Transactions on Energy Conversion* [online]. 2016, **31**(2), 520-530 [cit. 2017- 07-24]. DOI: 10.1109/TEC.2015.2514099. ISSN 0885-8969. URL: http://ieeexplore.ieee.org/document/7381688/

[2] DUMAN, Erkan, Hayrettin CAN and Erhan AKIN. FPGA based Hardware-in-the-Loop (HIL) simulation of induction machine model. 2014 16th International Power Electronics and Motion Control Conference and Exposition [online]. IEEE, 2014, , 616-621 [cit. 2017-07- 24]. DOI: 10.1109/EPEPEMC.2014.6980564. ISBN 978-1-4799-2060-0. URL: http://ieeexplore.ieee.org/document/6980564/

[3] MEWS, Marcus, Jaroslav SVACINA and Stephan WEILEDER. From AUTOSAR Models to Co-simulation for MiL-Testing in the Automotive Domain. 2012 IEEE Fifth International Conference on Software Testing, Verification and Validation [online]. IEEE, 2012, , 519-528 [cit. 2017- 07-24]. DOI: 10.1109/ICST.2012.137. ISBN 978-0-7695-4670-4. URL: http://ieeexplore.ieee.org/document/6200149/A

[4] "OpenModelica", *OpenModelica*, 2018. [Online]. Available: https://www.OpenModelica.org/. [Accessed: 13-Mar.-2018].

[5] "NI cRIO-9068", *Zone.ni.com*, 2018. [Online]. Available: http://zone.ni.com/reference/en-XX/help/373197K-01/cserieshelp/ni_9068/. [Accessed: 13-Mar.-2018].

[6] "Functional Mock-up Interface for Model Exchange and Co-Simulation", in *Fmi-standard*, 2018.